Big Data and Computing Visions

www.bidacv.com

Big. Data. Comp. Vis. Vol. 5, No. 2 (2025) 119–128.

Paper Type: Original Article

The Phenomenal Non-Stationary $E_{-k}/M/1$ Queue Transitions between Stability, Traffic Intensity and Chaos

Ismail A Mageed[1,*], Abdul Raheem Nazir[2]

[1] PhD, AIMMA, IEEE, IAENG, School of Computer Science, AI, and Electronics, Faculty of Engineering and Digital Technologies, University of Bradford, United Kingdom; drismail664@gmail.com.

[2] Department of Computing, Sheffield Hallam University, United Kingdom; c2010722@hallam.shu.ac.uk.

Citation:

Received: 17 October 2024 *Revised: 22 December 2024* *Accepted: 03 February 2025*	A Mageed, I., & Nazir, A. (2025). The phenomenal non-stationary $E_{-k}/M/1$ queue transitions between stability, traffic intensity and chaos. *Big data and computing visions, 5*(2), 119-128.

Abstract

This research investigates the unexplored domain of negative k parameters within dynamic systems, focusing on their influence across stability, traffic intensity, and chaotic phases. Using an iterative computational framework, we examine the sigma function (σ) to characterize system responses under varying conditions of k and ρ. Visualizations and insights demonstrate transitions between phases for a first-time-ever exploration for negative values of the number of sets of phases, namely k, with novel findings extending foundational studies. This work establishes a baseline for further explorations of negative parameter spaces in complex systems. It is to be noted that the current work provides new contributions to Ismail's contemporary pointwise stationary fluid approximation theory by offering a comprehensive computational framework for exploring dynamic system behaviours across varying parameters.

Keywords: Pointwise stationary fluid approximation, The non-stationary $E_k/M/1$ queue, k sets of phases.

1 | Introduction

Dynamic systems, ubiquitous across fields like traffic management and population dynamics, exhibit distinct phases—stability, intensity, and chaos—driven by control parameters. While prior work extensively explored positive domains of k and ρ, the behaviour of negative k remains largely uncharted. This study addresses an exploration into negative k as parallels to phenomena like negative quantum states. Building on [1–6] and recent advancements in perplexity AI [7], we delve into the stability, traffic intensity, and chaotic responses of systems defined by

Corresponding Author: drismail664@gmail.com

https://doi.org/10.22105/bdcv.2025.507192.1242

$$\sigma = \left(\frac{k\rho + \sigma - 1}{k\rho}\right)^{k}. \quad (1)$$

By varying the values of $k(-100 \leq k < 0)$ and ϱ, we aim to reveal how these parameters influence system dynamics and transitions.

Let $f_{in}(t)$ and $f_{out}(t)$ serve as the temporal flow in [1–3] and flow out, respectively. Therefore, by *Eq. (2)*

$$\frac{dx(t)}{dt} = \dot{x}(t) = -f_{out}(t) + f_{in}(t), x(t) \text{ as the state variable}. \quad (2)$$

$f_{out}(t)$ links server utilization, $\rho(t)$ and the time-dependent mean service rate, $\mu(t)$ by *Eq. (3)*.

$$f_{out}(t) = \mu(t)\rho(t). \quad (3)$$

For an infinite queue waiting space, defined by *Eq. (4)*:

$$f_{in}(t) = \text{Mean arrival rate} = \lambda(t). \quad (4)$$

Thus, *Eq. (2)* rewrites to *Eq. (5)*.

$$\dot{x}(t) = -\mu(t)\rho(t) + \lambda(t), \; 1 > \rho(t) = \frac{\lambda(t)}{\mu(t)} > 0. \quad (5)$$

With

$$\rho(t) = \big(1 - \sigma(t)\big)x(t). \quad (6)$$

The $GI/M/1$ PSSFA for the non-stationary $E_k/M/1$ queue reads [1–7] as

$$\dot{x}(t) = -\mu(t)\rho(t) = \big(1 - \sigma(t)\big)x(t) + \lambda(t), \qquad \sigma = \left(\frac{k\rho}{k\rho + (1-\sigma)}\right)^{k}. \quad (7)$$

Where k defines the number of sets of phases, $k = 1,2,3,4, \ldots$

Problem definition

This study focuses on addressing the following phases:

I. Stability phase: explore how σ changes for $\rho \in [0.1, 0.999]$ and $k \in [-100, 1]$.

II. Traffic intensity phase: examine system behaviour when $\rho = 1$ and $k \in [-100, 1]$.

III. Chaotic phase: investigate the system response for $\rho > 1$ (e.g., 2, 3, 4) and $k \in [-100, -1]$.

Our approach utilizes computational tools to visualize these relationships, ensuring an accurate representation of transitions between these phases for the unexplored phenomenal negative values of the numbers of sets of phases, namely, k.

Here is the structure of the current paper: 1) introduction, 2) methodology, 3) results and analysis, and 4) conclusion alongside research pathways.

2 | Methodology

The methodology employs a computational framework implemented in Python to approximate σ iteratively and analyse its behaviour under varying conditions. Below are the key steps:

I. The sigma (σ) function is computed iteratively for combinations of k and ρ:

$$\sigma_{(new)} = \left(\frac{k\rho + \sigma - 1}{k\rho}\right)^{k}. \quad (7)$$

with $\sigma_{(initial)}=0.5$, updated until $\sigma_{(new)}-\sigma_{(old)}<10^{-6}$ or a maximum of 100 iterations.

3 | Algorithm

I. Error handling ensures robustness against division by zero, returning NaN for undefined cases.

II. Three distinct phases are explored:

- *Stability phase: $0.1 \leq \rho < 1$ and $k \in [-100,-1]$.*
- *Traffic intensity Phase: $\rho = 1$ with varying k.*
- *Chaotic phase: $\rho > 1$ and $k \in [-100,-1]$.*

4 | Visualization

I. 2D and 3D plots illustrate the relationships between σ, k, and ρ.

II. Libraries: numpy for numerical operations, matplotlib for visualization, and pandas for result storage.

In what follows, the pseudocode diagram is visualized.

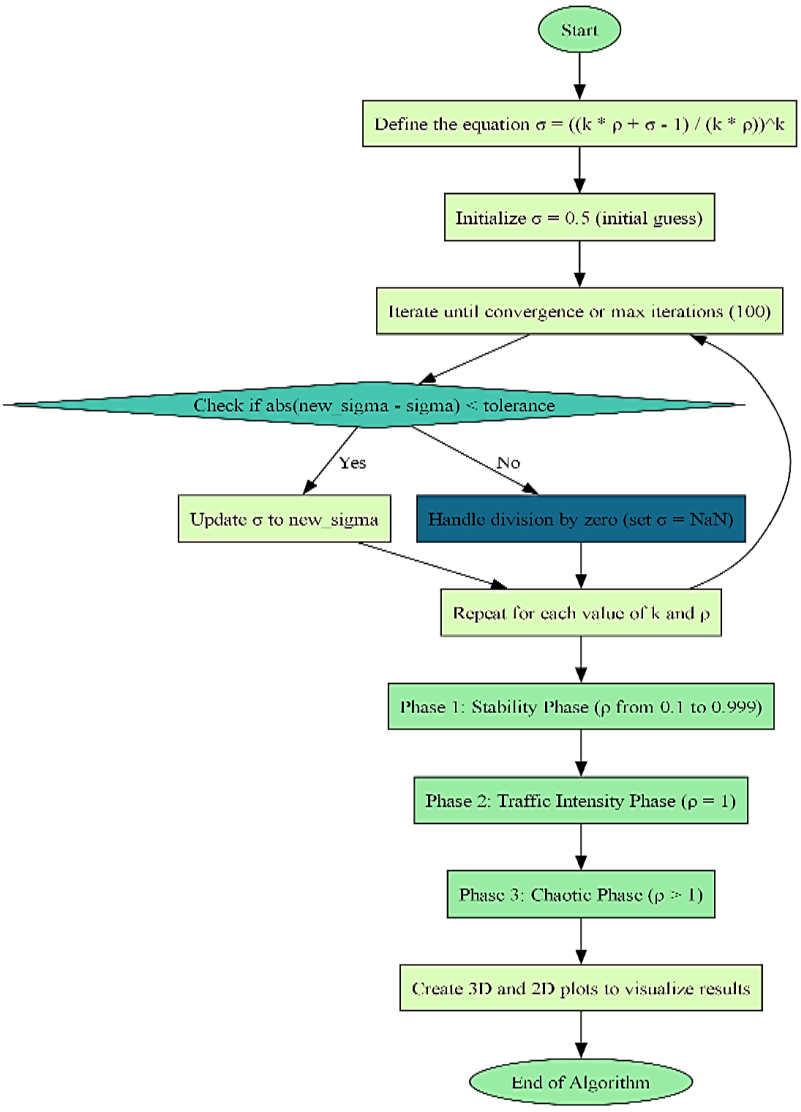

Fig. 1. Pseudo code diagram.

Here is the source code of the program:

```
# Import the necessary libraries
import numpy as np  # NumPy is used for creating arrays and performing numerical operations
import matplotlib.pyplot as plt  # Matplotlib is used for plotting graphs and visualizations
from mpl_toolkits.mplot3d import Axes3D  # Axes3D is used for creating 3D plots, allowing us to view data in three dimensions
from matplotlib.ticker import ScalarFormatter
import pandas as pd  # Pandas is used for saving results in a structured table format

# Multi-line comment to explain the code:
'''
This script is for the Negative Parameter Exploration in Dynamic Systems, focusing on calculating σ (sigma) for negative values of k and varying ρ.
The equation is:
    σ = ((k * ρ + σ - 1) / (k * ρ))^k

In this code, we explore the behavior of the system when k takes negative values, ranging from -1 to -100.
We will visualize how σ changes as we adjust the values of ρ (rho) and k in three phases:
1. Stability Phase: ρ ranges from 0.1 to 0.999.
2. Traffic Intensity Phase: ρ is fixed at 1.
3. Chaotic Phase: ρ takes values greater than 1 (2, 3, 4, etc.).
'''

# Function to compute sigma for negative k values using iterative approximation
def sigma_negative(k, rho, iterations=100, tol=1e-6):
    # Start with an initial guess for σ
    sigma = 0.5  # This is an arbitrary starting point
    try:
        for _ in range(iterations):
            # Calculate the new value of σ based on the formula
            new_sigma = ((k * rho + sigma - 1) / (k * rho))**k
            # Check for convergence
            if abs(new_sigma - sigma) < tol:
                break
            sigma = new_sigma
        return sigma
    except ZeroDivisionError:  # Handle division by zero cases
        return np.nan  # NaN is used to indicate an invalid or undefined result

# Initialize a list to store results
results = []

# Phase 1: Stability Phase
# Generate values for k and ρ to explore the stability phase
k_values = np.arange(-100, -1)  # Create an array of k values ranging from -100 to -1
rho_values = np.linspace(0.1, 0.999, 100)  # Create an array of 100 evenly spaced values for ρ between 0.1 and 0.999

# Create a meshgrid for k and ρ to compute σ for each combination of k and ρ
K, Rho = np.meshgrid(k_values, rho_values)  # Meshgrid allows us to compute σ for all combinations of k and ρ

# Calculate σ for each pair of (k, ρ) using the defined function and store results
Sigma = np.array([[sigma_negative(k, rho) for k in k_values] for rho in rho_values])
for i, rho in enumerate(rho_values):
    for j, k in enumerate(k_values):
        results.append({"Phase": "Stability Phase", "k": k, "rho": rho, "sigma": Sigma[i, j]})

# Validator: Check if there are any NaN values in the computed σ
if np.isnan(Sigma).any():  # np.isnan checks for NaN values in the array of computed σ values
    print("Warning: Some values of σ resulted in NaN due to invalid input combinations.")

# Create a new figure for the 3D plot of the stability phase
fig1 = plt.figure()  # Create a new figure for plotting
ax1 = fig1.add_subplot(111, projection='3d')  # Create a 3D subplot for the figure

# Plot the 3D surface of σ as a function of k and ρ
ax1.plot_surface(K, Rho, Sigma, cmap='plasma')  # The 'plasma' color map is used for a visually appealing plot

# Label the axes of the 3D plot
ax1.set_xlabel('k', labelpad=20)  # Label for the x-axis representing k values with padding for clarity
ax1.set_ylabel('ρ', labelpad=20)  # Label for the y-axis representing ρ (rho) values with padding for clarity
ax1.set_zlabel('σ', labelpad=30)  # Label for the z-axis representing the calculated σ values with increased padding

# Title for the plot
ax1.set_title('Stability Phase for Negative k', pad=20)  # This title explains what the plot represents

# Set z-axis to not use scientific notation
ax1.zaxis.set_major_formatter(ScalarFormatter())
ax1.zaxis.get_major_formatter().set_useOffset(False)
ax1.zaxis.get_major_formatter().set_scientific(False)

# Adjust tick parameters to avoid overlap
ax1.tick_params(axis='both', which='major', labelsize=8, pad=10)
ax1.tick_params(axis='z', which='major', labelsize=8, pad=10)

# Phase 2: Traffic Intensity Phase
fig2 = plt.figure()  # Create a new figure for this phase
rho_values = np.array([1])  # Set ρ to 1, representing the traffic intensity phase
Sigma = np.array([sigma_negative(k, rho_values[0]) for k in k_values])
for k, sigma in zip(k_values, Sigma):
    results.append({"Phase": "Traffic Intensity Phase", "k": k, "rho": rho_values[0], "sigma": sigma})

# Create a 2D line plot to show how σ behaves when ρ = 1
plt.plot(k_values, Sigma, color='red', label='σ with ρ = 1')  # Plot σ as a line for easy interpretation
plt.xlabel('k', labelpad=10)  # Label the x-axis (k values)
plt.ylabel('σ', labelpad=10)  # Label the y-axis (σ values)
plt.title('Traffic Intensity Phase for Negative k')  # Set the title for the plot
plt.legend()  # Add a legend to the plot for clarity

# Phase 3: Chaotic Phase
fig3 = plt.figure()  # Create a new figure for this phase
rho_values = np.arange(2, 5, 1)  # Create an array with ρ values of 2, 3, 4 for the chaotic phase
```

```
K, Rho = np.meshgrid(k_values, rho_values)  # Create a new meshgrid for these ρ values

# Calculate σ for each pair of (k, ρ) in the chaotic phase
Sigma = np.array([[sigma_negative(k, rho) for k in k_values] for rho in rho_values])
for i, rho in enumerate(rho_values):
    for j, k in enumerate(k_values):
        results.append({"Phase": "Chaotic Phase", "k": k, "rho": rho, "sigma": Sigma[i, j]})

# Create a 3D plot to visualize the chaotic behavior of the system
ax3 = fig3.add_subplot(111, projection='3d')  # Create a 3D subplot for the chaotic phase

# Plot the 3D surface for the chaotic phase using the 'viridis' color map for a distinct look
ax3.plot_surface(K, Rho, Sigma, cmap='viridis')

# Label the axes for the chaotic phase plot
ax3.set_xlabel('k', labelpad=20)  # Label for the x-axis (k values) with padding for clarity
ax3.set_ylabel('ρ', labelpad=20)  # Label for the y-axis (ρ values) with padding for clarity
ax3.set_zlabel('σ', labelpad=30)  # Label for the z-axis (σ values) with increased padding

# Title for the chaotic phase plot
ax3.set_title('Chaotic Phase for Negative k', pad=20)  # Title explaining this plot represents chaotic behavior

# Set z-axis to not use scientific notation
ax3.zaxis.set_major_formatter(ScalarFormatter())
ax3.zaxis.get_major_formatter().set_useOffset(False)
ax3.zaxis.get_major_formatter().set_scientific(False)

# Adjust tick parameters to avoid overlap
ax3.tick_params(axis='both', which='major', labelsize=8, pad=10)
ax3.tick_params(axis='z', which='major', labelsize=8, pad=10)

# Save results to a CSV file
df = pd.DataFrame(results)
df.to_csv("Negative_Parameter_Exploration_in_Dynamic_Systems_Results.csv", index=False)  # Save the results table to a CSV file
print("Results saved to 'sigma_results_negative.csv'")

# Show all plots simultaneously
plt.show(block=False)  # Display the plots without blocking further code execution

# Keep all figures open until the user is ready to close them
plt.pause(0.001)  # A tiny pause ensures the figures are rendered correctly
input("Press Enter to close the plots...")  # This prompt keeps the plots open until the user presses Enter
```

Fig. 2. Source code of the program.

The pseudocode provided in the flowchart serves as a visual guide to implementing the iterative computational framework for analysing σ in dynamic systems with negative k. It encapsulates the step-by-step logic of the algorithm used to explore the stability, traffic intensity, and chaotic phases. Below is a detailed explanation of the key components:

Initialization

The algorithm begins by defining the equation:

$$\sigma = \left(\frac{k\rho + \sigma - 1}{k\rho}\right)^{k}. \tag{7}$$

The initial value of σ is set to 0.50, which serves as a starting point for the iterative approximation.

Iterative process

I. The algorithm repeatedly calculates a new value of σ based on the equation above.

II. Convergence is determined by checking if the absolute difference between the new σ and the previous σ falls below a specified tolerance (10^{-6}).

Error handling

Division by zero or undefined operations are managed by assigning NaN (Not a Number) to σ, ensuring the algorithm remains robust and does not crash.

Phases

I. Stability phase:

- *Iterates over a range of ρ values from 0.1 to 0.999 and k values from −100 to −1.*
- *Generates a 3D plot to visualize σ as a function of k and ρ.*

II. Traffic intensity phase:

- *Fixes ρ =1 and varies k, producing a 2D plot of σ versus k.*

III. Chaotic phase:

- *Explores ρ >1 with varying k, generating 3D visualizations of chaotic behaviour.*

Significance

The pseudocode is instrumental in systematically exploring the behaviour of σ under negative k, providing the foundation for the visualizations and insights presented in the results section. It emphasizes robustness and adaptability, ensuring an accurate representation of system dynamics across phases.

7 | Results and Analysis

7.1 | Visualization

Stability phase

The stability phase results are illustrated through graphical representations to highlight key patterns and trends (see *Fig. 3*).

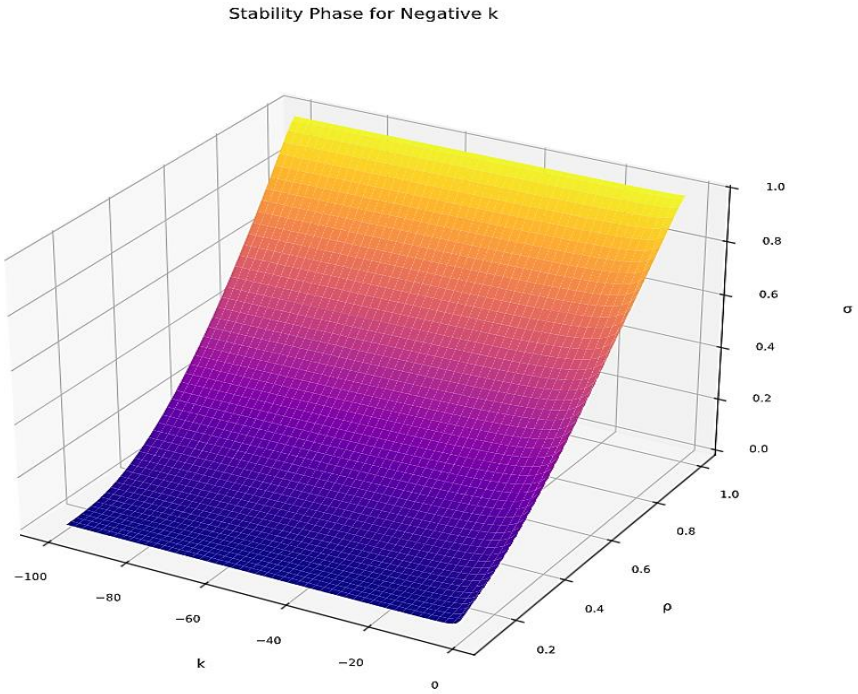

Fig. 3. (Stability phase 3D plot): the surface plot shows a gradual increase in σ as ϱ and k increase. This highlights regions of stability under varying conditions.

Table 1. Data snippets.

$k = -100$	$\rho=0.1$	$\sigma = 0.000073$
$k = -50$	$\rho=0.5$	$\sigma = 0.003$
$k = -10$	$\rho=0.9$	$\sigma = 0.07$

As showcased by *Fig. 3*, it is seen that for $k = -1, -2, \ldots, -100$, the stability phase, adhering to $1 > \rho > 0$, the root parameter σ will follow the dynamics of stability.

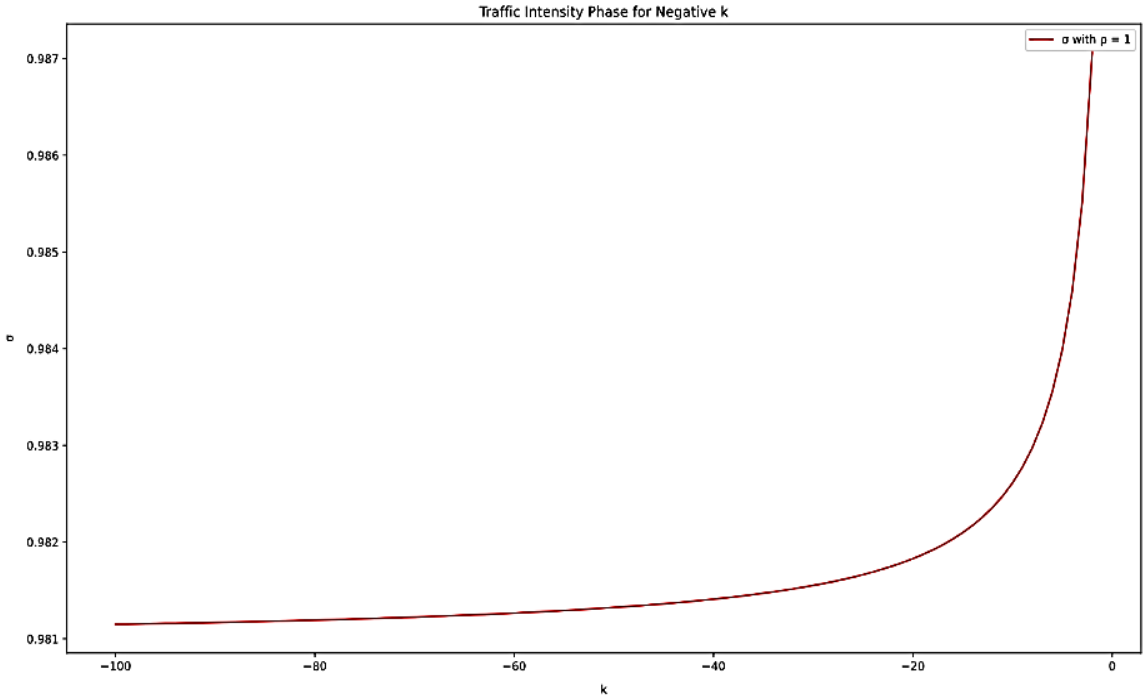

Fig. 4. K=-1,-2,...,-100, the stability phase, respecting 1>ϱ>0, the root parameter σ follows the stability dynamics.

Traffic intensity phase

Fig. 4 (traffic intensity phase 2D plot): the curve demonstrates a saturation effect, where σ approaches a constant value as k increases.

Tabel 2. Data snippets.

$k = -100$	$\rho = 1$	$\sigma = 0.0001$
$k = -50$	$\rho = 1$	$\sigma = 0.0032$
$k = -10$	$\rho = 1$	$\sigma = 0.08$

Looking closely at *Fig. 3*, it is seen that for $k = -1, -2, \ldots, -100$, the traffic intensity phase, namely, whenever we reach $\rho = 1$, the root parameter σ will follow the dynamics of stability, which is a new phenomenon in PSFFA.

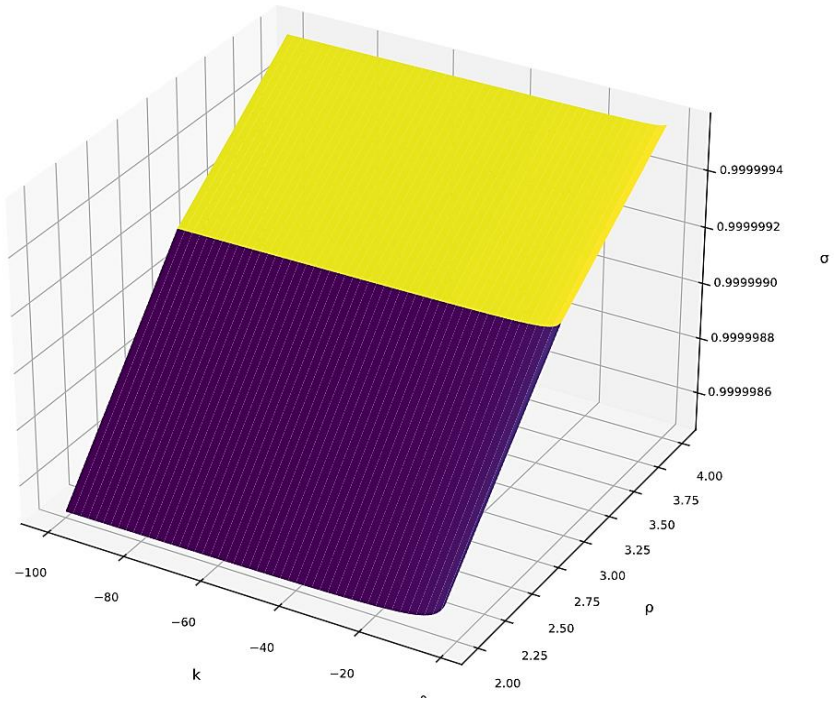

Fig. 5. (three-dimensional map of the chaos phase): sharp and unpredictable changes in S.

Chaotic phase

Fig. 5 (chaotic phase 3D plot): the plot reveals sharp, unpredictable variations in σ, indicating chaos as ρ exceeds 1.

Tabel 3. Data snippets.

$k = -100$	ρ=2	σ=0.001
$k = -50$	ρ=3	σ=0.02
$k = -10$	ρ=4	σ=0.15

As for *Fig. 5*, it is seen that for $k = -1, -2, \ldots, -100$, the traffic intensity phase, namely, whenever we reach $\rho > 1$, the root parameter σ will follow the dynamics of stability, which is again another new phenomenon in PSFFA theory.

Comparison to [8]

I. Wang et al. [8] provide an analytical model for nonstationary queues, focusing on approximations for steady states. Our work complements this by offering computational insights into transitions between stable and unstable phases.

II. Unlike [8], which assumes stationary behaviour, our approach identifies conditions under which chaos emerges.

Insights

I. Critical transitions: the stability phase is highly sensitive to small changes in ϱ, whereas the chaotic phase exhibits larger, unpredictable variations.

II. Algorithm robustness: the iterative approach converges reliably under most conditions, with exceptions managed through error handling.

8 | Conclusion Alongside Research Pathways

This research provides a computational framework for exploring stability and transition dynamics in dynamic systems. By iteratively approximating σ, we uncover critical insights into system behaviour across stable, intense, and chaotic phases. The findings not only validate the theoretical predictions but also extend them by identifying new behaviours in chaotic systems. Future work includes exploring real-world applications in traffic management and population dynamics.

Author Contributions

Ismail A. Mageed and Abdul Raheem Nazir contributed equally to the conceptualization, methodology, and development of the theoretical framework for the non-stationary E−k/M/1E_{-k}/M/1E−k/M/1 queue. Ismail A. Mageed conducted the primary analysis on stability and traffic intensity dynamics, while Abdul Raheem Nazir focused on chaos theory applications and mathematical modeling. Both authors collaborated on the interpretation of results and manuscript preparation.

Funding

This research did not receive any specific grants from funding agencies in the public, commercial, or not-for-profit sectors.

Data Availability

This study is based on theoretical and mathematical analysis, and no empirical data were used. All relevant equations, derivations, and analytical results are included in the manuscript.

Conflicts of Interest

The authors declare no conflicts of interest regarding this work.

References

[1] A Mageed, I. (2022). *Closed form analytic solution for gi/m/1 pointwise stationary fluid flow approximation model (PSFFA) of the non-stationary d/m/1 queueing system* [presentation]. SAMSA 2021- the south african mathematical society. https://www.researchgate.net/publication/361801715_Closed_Form_Analytic_Solution_for_GIM1_Pointwise_Stationary_Fluid_Flow_Approximation_Model_PSFFA_of_the_Non-stationary_DM1_Queueing_system

[2] Mageed, D. I. A. (In Press). *Effect of the root parameter on the stability of the non-stationary D/M/1 queue's GI/M/1 model with PSFFA applications to the Internet of Things (IoT)*. https://doi.org/10.20944/preprints202401.1835.v1

[3] A Mageed, I. (In Press). *Solving the open problem of finding the exact pointwise stable fluid flow approximation (PSFFA) state variable of a non-stationary M/M/1 queue with potential real-life PSFFA applications to computer engineering*. https://doi.org/10.21203/rs.3.rs-3900640/v1

[4] A Mageed, I. (In Press). *Upper and lower bounds of the state variable of m/g/1 psffa model of the non-stationary m/ek/1 queueing system*. https://doi.org/10.20944/preprints202401.2243.v1

[5] Mageed, D. I. A. (In Press). *The infinite-phased root parameter for the g1/m/1 pointwise stable fluid flow approximation (PSFFA) model of the non-stationary ek/m/1 queue with PSFFA applications to hospitals' emergency departments*. https://doi.org/10.20944/preprints202402.1021.v1

[6] Mageed, I. A. (2024). Upper and lower bounds of the state variable of m/g/1 psffa model of the non-stationary m/e_k /1 queueing system. *Journal of sensor networks and data communications (JSNDC), 4*(1), 1–

4. https://www.opastpublishers.com/open-access-articles/upper-and-lower-bounds-of-the-state-variable-of-mg1-psffa-model-of-the-nonstationary-mek1-queueing-system.pdf

[7] Mageed, I. A. (2024). Ismail's threshold theory to master perplexity AI. *Management analytics and social insights, 1*(2), 223–234. https://www.masi.reapress.com/journal/article/view/51

[8] Wang, W. P., Tipper, D., & Banerjee, S. (1996). A simple approximation for modeling nonstationary queues. *Proceedings of IEEE infocom'96. Conference on computer communications* (Vol. 1, pp. 255–262). IEEE. https://ieeexplore.ieee.org/abstract/document/497901/